Monitor Your PATH and LD_LIBRARY_PATH to Avoid Software Conflicts

When building or running an application, the environment variables PATH and LD_LIBRARY_PATH may be used to find the commands and/or runtime shared libraries needed for the operation. If multiple entries in these variables contain commands or libraries with the same filename as the one needed, this can potentially lead to conflicts.

The system searches the PATH or LD_LIBRARY_PATH entries from beginning to end, and always uses the first matching entry it finds—regardless of whether it is the correct one. Using the wrong command or shared library could cause your application to fail or provide incorrect results. Therefore, it is very important to pay attention to which entries are included in PATH and LD_LIBRARY_PATH, and where they are placed.

PATH and LD_LIBRARY_PATH Entries

Some default directories (such as /usr/local/bin and /usr/bin) are automatically included in PATH for every NAS user account. Many users modify PATH in their system startup files (such as .cshrc, .login, .profile, or .bashrc). For LD_LIBRARY_PATH, a system default is not defined; users typically define LD_LIBRARY_PATH in their own scripts.

Software modules add entries to PATH and LD_LIBRARY_PATH when they are loaded. The placement of an entry depends on two factors:

- Whether a modulefile uses prepend-path or append-path.

  For example, the tecplot/2017r2 modulefile prepends the /nasa/tecplot/2017r2/360ex_2017r2/bin:/nasa/tecplot/2017r2/chorus_2017r2/bin entry to the beginning of the existing PATH, while the netcdf/4.4.1.1_serial modulefile appends the /nasa/netcdf/4.4.1.1_serial/bin entry to the end of the PATH.

  To find out whether prepend-path or append-path is used by a given modulefile, run:

  module show *modulefile_name*

- The order in which the modulefiles are loaded.

  Entries are added to PATH and LD_LIBRARY_PATH in the order the modulefiles are loaded. For example, both the hdf5/1.8.18_serial and pkgsrc/2016Q4 modulefiles use prepend-path to add their corresponding bin directories to PATH. If hdf5/1.8.18_serial is loaded before pkgsrc/2016Q4, the entries from pkgsrc will be listed first, as follows:

  ```
  /nasa/pkgsrc/sles12/2016Q4/bin:/nasa/pkgsrc/sles12/2016Q4/sbin:/nasa/hdf5/1.8.18_serial/bin
  ```

Known Conflicts

It is a common problem that users load a long list of modules for their entire workflow in their system startup scripts, but are not aware that some of these modules contain commands and/or shared libraries that are in conflict with one another. A few known conflicts are described below.

NAS pkgsrc Module

The NAS-built pkgsrc module contains nearly 1,000 packages. Its bin and lib directories contain many commands and libraries that conflict with the bin and lib directories in the system default locations or other software modules. For example:

- The cmake command is included in both the /usr/bin and /nasa/pkgsrc/sles12/2016Q4/bin directories.
- The libhdf5.so library is included in the /nasa/pkgsrc/sles12/2016Q4/lib directory, the /nasa/hdf5/1.8.18_serial/lib directory, and the /nasa/hdf5/1.8.18_mpt/lib directory.

Other Software Packages

Some commercial or third-party software packages are self-contained and might include duplicate commands or libraries that are available in other modulefiles. For example:

- The Tecplot package comes with its own Intel libraries, such as libimf.so, libifcore.so, and libifport.so. If your LD_LIBRARY_PATH includes both Tecplot's library directories and the Intel compiler's library directories, your application could fail in unpredictable ways—or worse, it could silently return incorrect results.
- Many commercial packages (such as matlab, ansys, flow3d, ansa, nastran, star-ccm, mathematica, and so on) provide the libiomp5.so library in their distributions. Loading one of these software modules together with an Intel compiler module might cause problems for OpenMP applications.

  For example, running the OVERFLOW MPI+OpenMP hybrid code results in the following error when a matlab module is loaded after an Intel compiler module:

  ```
  overflowmpi: relocation error: overflowmpi: symbol kmp_aligned_malloc, version VERSION not defined in file libiomp5.so with link time reference
  ```

  This occurs because the matlab modulefile prepends its library path to LD_LIBRARY_PATH, so that entry is listed first. The libiomp5.so library needed by OVERFLOW is therefore found in the matlab distribution's entry (wrong copy) instead of in the Intel distribution's entry (correct copy).

Best Practices

To avoid software conflicts, follow these best practices:

- Load modules in a working session and/or in PBS scripts, instead of in your shell startup scripts.
- Do not load the pkgsrc module unless it is needed.
- Load only the modules that are needed by the application you are currently using.
- When in doubt, check which command or library you are using.

  To find the directory containing the command you are using, run:

  ```
  which command_name
  ```

  To find the paths to shared libraries satisfied by the command you are using, run:

  ```
  ldd directory/command_name
  ```

Article ID: 547
Last updated: 07 Dec, 2017
Updated by: Moyer M.
Revision: 61
Filesystems & Software -> Monitor Your PATH and LD_LIBRARY_PATH to Avoid Software Conflicts
https://www.nas.nasa.gov/hecc/support/kb/entry/547/